VirtualFriendship: Hiding interactions on Online Social Networks

Anonymous Submission

ABSTRACT

The enormous popularity of Online Social Networks (OSNs) dramatically changed the way people communicate and interact with each other. Unfortunately, the centralized control employed by OSN providers, such as Facebook, have negative impacts as it leads to privacy concerns. In fact, OSN providers can directly access a big amount of information about users and infer further information when using data mining techniques (e.g., about the on line interactions among two users).

In this paper we propose VirtualFriendship, i.e., a collaborative solution that allows OSN users to browse each others' "profile" and to exchange messages in the OSN, while the end-peers of the interactions remain anonymous with respect to the OSN provider. We demonstrate by a proof of knowledge implementation (VF-App) the feasibility of our solution. Furthermore, our analysis and thorough set of experiments also show that the overhead introduced by our solution is very limited, and we believe negligible from a user perspective.

Categories and Subject Descriptors

C.2.0 [**Computer Communications Networks**]: General—*Security and protection*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

Keywords

Online Social Networks, privacy, anonymity

1. INTRODUCTION

Online Social Networks (OSNs), like Facebook, Google+, and LinkedIn, have conquered current society and become an important communication channel for million users. In fact, OSNs offer an easy way to share information and keep updated with friends' activities. Naturally, OSNs implicitly became the depository of large quantities of users shared information. They are provided with the means to extract and access huge amounts of sensitive information, either directly or through data mining techniques. In this way, OSNs can easily infer user's closest friends (e.g., by mining the number and type of interactions among them), interests, political and social orientation. A recent example of possible collection and mining over users data is the NSA PRISM project [35], apparently authorized from US federal judges overseeing the Foreign Intelligence Surveillance Act (FISA). In addition, providers may grant advertisement companies access without the user's consent for economical reasons [36]. Inherently, all these worrisome issues lead to the increase of privacy and security concerns among users [22].

Currently, most OSN providers make some customizable "privacy settings" to its users, so that each user can employ some levels of access control on his published content. Whilst such mechanism provides users with a certain level of control, it also relies on the trustworthiness of the OSN providers in managing their content and enforcing access control rules. Hence, the most effective solution for privacy enforcement in OSNs is to give full control to the users or to build new OSN systems more effective in protecting users privacy. Towards this goal, several solutions have been proposed: some rely on cryptographic techniques [2, 5, 7, 25, 31]; and some on new proposed systems mostly peer-to-peer based [15, 21]. However, even if content is kept confidential from the prying eyes of unwanted viewers, the interactions between users may disclose sensitive information [8, 28, 41]. Also, for peer-to-peer based solutions [15, 21], the availability of information represents an concerns. Furthermore, the tradeoff of moving to a new OSN provider and lose the interaction with potentially less privacy-concern friends is high.

Most solutions, however, tackle the problem of privacy as confidentiality of the content. In fact, they aim to hide the content from un authorized viewers, with main focus on the OSN provider. Nonetheless, as observed in [3] the OSN provider is still able to learns and to extract extra sensitive information based on users behavior, such as the strength of relationships, by employing data mining techniques. Hence, in this paper, we address a specific class of privacy concerns, i.e., where sensitive information can be inferred from the behavior and connections of users. For example, if Alice accesses Bob's profile very often, just because Bob shares information in line with Alice's interests. Then, the OSN could classify Bob as Alice close friend. In order to avoid such leakage, Alice needs to access Bob's information in such a way that the OSN is kept oblivious. Usually, anonymous networks, such as Tor [19], are used as the solution to hide identity of the communication and achieve anonymity. How-

ever, users in OSNs are required to be log in, and thus, such solutions would not work.

In this paper we propose, to the best of our knowledge, the first hybrid system approach, named VirtualFriendship (or VF for short). In this way, there is no need to re-design current OSNs nor advocate the move to new freshly privacy-friendly OSNs. Our solution, allows users to keep the information stored in any centralized OSN, while communicating anonymously. To do so, VirtualFriendship performs all private communications using a decentralized peer-to-peer system, where each peer is represented by a user in the system. To achieve anonymity and unobservability, we present the novel concept of *routing friends*, where users abuse the social trust, as defined in [17], of their OSN connections to route and process requests. Also, the *routing friends* channel the communication through an anonymous network, such as Tor, to conserve the requester identity. In this way, users communicate privately towards a decentralized channel by keep all the benefits of storage of centralized OSNs.

To demonstrate the feasibility of VirtualFriendship, we implemented it as a proof-of-concept (VF-App) in the form of a Firefox extension. Further, we have analyzed the VF-app performance and practicality when using Facebook.

Contributions: This paper makes several contributions. First, it presents a hybrid solution to protect OSN user's anonymity with respect to the OSN provider. In particular we anonymize user when accessing information of other users, keeping at the same time also the real friendship weight oblivious. Second, our solution also allows two OSN users to communicate in a privacy friendly way. To the best of our knowledge, this is the first solution that achieves these goals while preserving current OSN functionalities (e.g., in our particular implementation for Facebook). In the process, we identify several open issues and future research directions. Finally, we build and evaluate an open-source prototype.

Outline: The rest of this paper is organized as follows: Section 2 surveys the related work. Section 3 describes our system, adversarial models and sketches the privacy goals. Section 4 formalizes the architecture and operations of our system, while Section 5 analyses the security. We describe our prototype implementation in Section 6. Later, Section 7 presents a short discussion on possible extensions. Finally, Section 8 concludes the paper.

2. RELATED WORK

In this section we review some related work that addresses different privacy issues in OSNs. With the increase popularity of OSNs several privacy concerns start to arise which have prompted a large interest within the research community. A number of studies [9, 20, 23, 27] enumerate privacy issues and challenges in OSNs. Recently, [1] reviewed the state of the art in privacy OSN and defined several open questions.

Most proposed solutions focus on protecting user's published content information which is directly available to OSN providers, such as profile information, status updates, comments, and images. For instance, solutions like FaceVPSN [14] and NOYB [25] protect Facebook users profile information by using fake information. While systems such as [2, 5, 7, 31] make use of cryptographic mechanisms to enforce confidentiality and access control to the published content shared by the users. However, even if the content is protected using such solutions, such solutions do not consider the anonymity issue, i.e., an adversary can still infer sensitive information from the interactions. For example, Facebook, already uses such information to display the top friends of each user on the right hand side of the main page, and also on the chat and content likes order.

In [3] the authors address a different, yet equally important, class of privacy issues in OSNs: the fact that sensitive information can be inferred by user behavior. They study and analyze the privacy guarantees with respect to user anonymity, considering users introducing dummy traffic in OSNs. The issue of this solution is the extended storage overhead with amount of information that each user needs to generate and dump into the OSN.

Other solutions take a more drastic approach proposing new systems to replace existing OSNs: examples of such solutions are Diaspora [21] and Safebook [16], mainly based on decentralized architecture. Unfortunately, these solutions rely not only on the availability of peers but also on the assumption that a large mass of users move to a new system, in detriment to existing ones where all their friends are.

In contrast to previous solutions, the proposals in [17, 32] use OSN graph as mixers to provide anonymity. Drac [17] proposes an architecture to provide anonymity and unobservability for low-volume communications, such as chat messaging. Pisces [32], on the other hand, proposes a decentralized anonymous communication protocol as an alternative to Tor [19]. Both make use of the social trust of OSN friendships to initiate the circuit to relay anonymous traffic. As result, Dynamix [33] analyses how anonymity is affected when the social graph is dynamic. However, such solutions present a different threat model and do not aim to address OSN concerns. Furthermore, due to the fact that users require to be registered and logged in when using OSNs, the direct use of such system does not really help to protect users anonymity towards the OSN provider.

3. MODEL

In this section, we explain our system model and assumptions (Section 3.1), as well as the adversarial model we consider in this paper (Section 3.2). We conclude by sketching the main privacy goals this work aims to solve (Section 3.3). In Table 1, we summarize the notations used throughout the paper.

3.1 System Model

We consider a system where the OSN is centralized (e.g., Facebook) and follows the OSN definition from [9], such that: users create profiles $\mathcal{P}$, have a list of friends $\mathcal{R}$ with whom have relationships, share interests and a messages m. Without loss of generality, throughout the rest of the paper, we consider two users of an untrusted OSN: Alice and Bob, such that, $\mathcal{P}_{Alice}$ represents Alice's profile and $\mathcal{R}_{Alice}$ her list of friendship relationships in the OSN. We stress that $\mathcal{P}_{Alice}$ information is private and only accessible by her friends in $\mathcal{R}_{Alice}$, and by the OSN provider.

Besides all the traffic that Alice can perform when using a OSN, we generalize and assume that Alice performs mainly the following three actions: (1) accesses Bob's private information $\mathcal{P}_{Bob}$; (2) exchanges messages m with Bob; and (3) post comments. We assume that all exchanged messages are encrypted and hidden from the OSN provider or other

prying eyes. In addition, we note that on our system, Alice is not required to be a registered member of the OSN to retrieve Bob's information.

Table 1: Notation

Symbol	*Meaning*
u	Identity of the users (e.g., Alice and Bob)
m	content exchanged among users (e.g., comment, post)
$\mathcal{P}_u$	Profile of the user u
$\mathcal{R}_u$	Relationships of the user u
τ_u	Authorization token for $\mathcal{P}_u$
$\mathcal{L}_u^l$	List or group of users, with *label* l, defined by u
Λ_u	User u local server
$\mathcal{F}_u$	Routing Friend of u, used as connecting point to u's information (e.g., $\mathcal{P}_u$ or m)
$\Gamma^n_{(u,u')}$	List of size n containing the tuples ($\mathcal{F}_{u_i}$,Λ_i) that u can use to access content from u'
(pk_u,sk_u),κ	Asymmetric key pair (public and private keys), and symmetric key from user u

3.2 Adversarial Model

Throughout the rest of the paper we will consider a passive adversary that monitors the user traffic in the OSN (e.g., OSN provider), and is able to access users' content, the list of friends of every single user in the OSN and the interactions between them. We infer that such adversary adheres to the honest-but-curious model by following with the protocol specification and not tampering with content. However, such adversary aims to eavesdrop, track and recover information of the communication, in such a way that he can compute (for example, by using probabilistic methods) the weight matching the importance of friendship connections and retrieve main interests. For instance, a naive approach on computing the friendship strength between two users is to consider that each interaction is of power one and the final strength is the sum of all interactions. Also, the adversary should be kept oblivious of the real shared content.

We assume that direct social relationships $\mathcal{R}$ are trusted, i.e., that a single user trust his friends as the entry and exit points of the relay circuit. However, we consider that the exit points of the relay system do not deviate from the protocol specification, modeled as honest but curious. Furthermore, we assume that all communications are encrypted, and that the profile information can be concealed using methods like FaceVPSN [14] or NOYB [25]. Although, it is assume that OSNs do not allow publish of encrypted content, we argue that this ca be bypassed as in [5, 31]. Finally, we stress the fact that side channels attacks, such as timing and correlation attacks, are beyond the scope of this paper.

3.3 Privacy Definitions and Goals

Privacy in OSNs have lead to discussion and interests by the research community and the media. Yet, the difficulty of the problem is far from solved. In fact, providing a definition on privacy in a OSN system is a challenging task that we do not aim to solve within this paper. As defined in [26], privacy can be defined as different paradigms. We mainly focus on privacy as confidentiality and the right to hide sensitive information. However, such paradigm can also be divided into different categories, due to the complexity of information shared in OSNs. Therefore, in this paper we consider such categories as our privacy goals and we divide them as follows:

User Anonymity: we aim that Alice is not identifiable towards other entities (i.e., OSN provider and other users, inclusive Bob), when accessing Bob's profile or other content in the OSN. In this way, the real friendship weight as defined in [3] is not given to an adversary. In addition, Alice's interests are also kept oblivious to the adversary as its traffic is routed.

Communication Anonymity: we target that the communication among two users is unobservable. In other words, that it is hard to detect that Alice and Bob are exchanging messages.

Content Privacy: to guarantee that the content exchanged or published is kept confidential. The content can be entrusted with a limited of recipients by creating access lists. Also, the system should guarantee the integrity of publishing content to avoid impersonation attacks.

In addition, the system should be deployable and the design of the system should be an hybrid design, i.e., it should keep the central structure and general functionality of the OSN, providing availability of data. While the private communication should be done using a decentralized peer-to-peer system, where each peer is represented by a user in the system.

4. VIRTUALFRIENDSHIP SYSTEM

In this section we start by sketching an overview of the architecture, followed by a full detailed description of the proposed solution that this work addresses.

4.1 Architecture

Our system is a hybrid system; each user u in the system besides of being represented in the OSN with $\mathcal{P}_u$ it also runs a local server Λ_u that allows to relay the traffic outside the centralized control of the OSN. The collection of the local servers Λ_u can be seen as an additional independent decentralized network. The local servers Λ_u are used to route users' traffic, such as, establish connections, extract profiles and make comments. Along with an anonymous network (AN), such as Tor, users can perform the actions aforementioned in an anonymous way. In addition, they also verify if the user requesting the information is allowed to access it. For instance, if Alice when requesting $\mathcal{P}_{Bob}$ contains the authorization token τ_{Bob}.

The architecture of our system involves three entities: Users, the OSN and an Anonymous Network (AN).

Users: In the system we consider two types of users: (1) the *communication users*, that are the users exchanging information, e.g., Alice and Bob, where Alice requests Bob's profile; (2) the *routing users/friends* $\mathcal{F}$, are regular users that act as the entry and exit points to the requested information on the OSN (e.g., $\mathcal{P}_{Bob}$). More formally, exit points operate the request from the direct connection (e.g., Alice) to another routing friend from the destination (e.g., Bob) throughout an anonymous network (e.g., Tor). Whereas the entry points

authenticate and retrieve the requested content (e.g., $\mathcal{P}_{Bob}$). In fact, each user in the system runs a local server Λ that is distinguished by a unique identifier (e.g., Facebook username), and holds an asymmetric key pair (pk, sk). The key pair is generated when the system is deployed and could be made available on users profiles, for example, using a QR code image.

OSN: We assume the OSN to be any centralized OSN, such as Facebook or Google+, and to be the base communication channel. Also, provide the availability of the information.

Anonymous Network (AN): Tunneling through an AN provides attractive security and privacy features. While the most marked is enhanced anonymity, it also offers encryption. For our system, the anonymous network is used to provide anonymity to the content requester (e.g., Alice when requesting $\mathcal{P}_{Bob}$). This can be any centralized or decentralized AN, for instance, Tor [19] or Tarzan [24] respectively. Although we assume the use of Tor throughout the paper, the definition of a AN goes beyond this research paper.

4.2 System Overview

We propose a hybrid architecture that allows users to communicate through any centralized OSN in an privacy friendly way. With such solution there is no need to re-design current OSNs nor advocate the switch to a new freshly designed OSN. In contrast, users use an extra entities that allows them to route traffic through a AN. This is done in a similar fashion as [17, 19, 32]. In this way, users keep the main information stored in the OSN, which should be in encrypted format [2, 5, 7, 31] or other fake information such as in [14, 25]. For instance, the case where Alice is a user of a OSN that wants to perform a OSN action such as retrieving the profile information from another OSN user Bob, in such a way that the OSN is kept oblivious on the fact that Alice accessed it. Note that Alice may be an independent outside friend of Bob, i.e., as she is not required to be registered to the OSN.

Usually, since Alice needs to be signed in to the OSN to use it, there will be no point to route the traffic from inside the network. Thus, using an anonymous network (AN), such as Tor, to browse the OSN would not be a valid option. Due to this Alice's communications are required to be routed outside the OSN. To do so, we introduce the concept of *routing friends* $\mathcal{F}$. Such friends act as intermediaries for Alice – Bob communication. They receive requests and redirect them using a AN to other routing friends.

In order to create an anonymous channel of communication Alice abuses the routing friends along with an AN. To do so, we use Tor to create a tunnel between Alice - Bob using routing friends $\mathcal{F}_{Alice_i}$ and $\mathcal{F}_{Bob_j}$. The usage of $\mathcal{F}_{Alice_i}$ is to protect Alice's identity (e.g., Facebook ID or IP address) when accessing the AN, which replaces Alice's identity by its own on the request. This occurs because in Tor users need to place trust on the entry point, thus, we follow the definition of social trust from [17]. As prior defined in our model, we motivate the use of an AN on the fact that Alice does not necessary trust the $\mathcal{F}_{Bob_j}$ from the set $\Gamma^n_{(Bob,Alice)}$ revealing his identity to the OSN provider or any other third party.

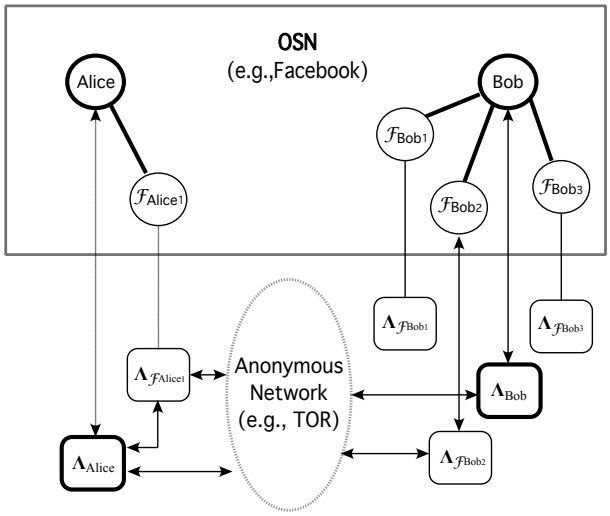

Figure 1: System Overview

To summarize, our system key design elements are as follows:

- Users perform content requests, exchange encrypted messages, and publish information.
- Routing friends operate, re-direct requests and check authorization tokens.
- Only authorized users can access and decrypt requested content.

Figure 1 illustrates an overview of our system, where Alice accesses Bob's profile $\mathcal{P}_{Bob}$ using $\mathcal{F}_{Bob_2}$, such that, $\mathcal{F}_{Bob_2} \in \mathcal{R}_{Bob}$. The connection between Alice and $\mathcal{F}_{Bob_2}$ is performed using Λ_{Alice} and $\Lambda_{\mathcal{F}_{Bob_2}}$ that are connected through a AN. Note that, $\mathcal{F}_{Bob_2}$ is a *routing friend* and is required to be online to deal with Alice's request. Also, Alice's request can also be performed directly to Bob. For users that only trust their direct connection, and as in systems like Tor the first entry point is required to be trusted, Alice can make her request through her routing friend $\mathcal{F}_{Alice_1}$.

4.3 System Operations

In this section, we overview our system by describing the operations users are able to perform. We highlight the fact that we wish to provide the same main functionalities as the OSN. However, in this work we does not address voting actions, such as "like" in Facebook. We start by explaining the initialization of the system, followed by the profile extraction, and concluding with the handling of comments and posts. We consider that both Alice and Bob are registered and represented in the OSN by $\mathcal{P}_{Alice}$ and $\mathcal{P}_{Bob}$, respectively. However, the process does not change if Alice is an independent user not registered to the OSN.

Initialization

To bootstrap the system, Alice and Bob need to become connected, in such a way that, Bob $\in \mathcal{R}_{Alice}$ and Alice $\in \mathcal{R}_{Bob}$. When Alice and Bob establish a connection, they exchange an initial set of values ($\mathtt{I}_u$): composed by a list of routing friends along with their Λ information and an authorization

token τ. Whereas, the τ is a random value used for user authentication, the $\mathcal{L}$ represents the access groups pre-defined by the user. Note that, $\Gamma_{(,)}$ is a sub set of $\mathcal{L}$ as not all users in $\mathcal{L}$ act as a routing friend. Currently, the token τ is a long lived token, however, it can become short lived when associated with a timestamp. This option, however, increases the communication overhead. Actually, Alice and Bob can specify a τ per group $\mathcal{L}$, assuming the anonymity set is large enough (i.e., the amount of users in the group is such that is hard to identify which one is accessing the content). Let $\texttt{Enc}_{pk}(\cdot)$ and $\texttt{Sign}_{sk}(\cdot)$ represent an asymmetric encryption and a digital signature algorithms, respectively, and $\{\cdot\}_{\kappa}$ a symmetric authenticated encryption (e.g., AES in CCM mode [40], or a dedicated scheme [42]), and $H(\cdot)$ a collision resistant hash function. The initialization protocol, reported in Figure 2, is a two-step protocol initiated by Alice. First, Alice encrypts (using pk_{Bob}) $\texttt{I}_{Alice}$ along with a random nonce N, and add a signature for $\texttt{I}_{Alice}$. Bob, on the other hand, decrypts and verifies the authenticity of the content. Then, Bob replies with the encryption (symmetric, for efficiency reasons) of $\texttt{I}_{Bob}$. Note that, if Bob specifies groups, such that, for example, $Alice \in \mathcal{L}^{Work}_{Bob}$, then all members in $\Gamma^{n}_{(Alice,Bob)}$ are also in $\mathcal{L}^{Work}_{Bob}$. Subsequently, the overhead storage of τ from each connection (e.g., Alice storage of τ_{Bob}) is linear with the number of connections, whereas, the own τ (e.g., Alice manage τ_{Alice}) overhead is either linear with the number of groups. For revocation, Alice re-generates a new τ and shares with the connections with the affected connections, either all or from a specific group. This, however, creates a linear communication overhead.

Currently, this process is operated between Λ_{Alice} and Λ_{Bob}, using the unique identifiers as the local server addresses. However, as it is performed in encrypted format, this process can be executed using either an out-of-band communication channel (e.g., email) or directly in-band (e.g., inside the OSN).

Let $\texttt{I}_u \leftarrow \{pk_u, \tau_u, \Gamma^{n}_{(u,u')} \leftarrow [(\mathcal{F}_{u_{x_i}}, \Lambda_{x_i}), ..., (\mathcal{F}_{u_{x_n}}, \Lambda_{x_n})]\}$ for $u = \{Alice, Bob\}$

Alice $\rightarrow$ Bob: $\texttt{Enc}_{pk_{Bob}}(I_{Alice}||N), \texttt{Sign}_{sk_{Alice}}(I_{Alice})$

Alice $\leftarrow$ Bob: $\{I_{Bob}\}_{\kappa}$, s.t. $\kappa = H(\tau_{Alice}||N)$

Figure 2: Initialization protocol between two users – Alice and Bob.

A different approach, yet less efficient but forward secure, is to use anonymous credentials [10, 12], where the τ represents the proof of knowledge of a credential issued by Bob. For instance, Bob issues a credential Alice for accessing $\mathcal{P}_{Bob}$, such that, when Alice requests $\mathcal{P}_{Bob}$, she computes τ as the proof of knowledge of the credential.

Accessing Content

In order to access Bob's Profile ($\mathcal{P}_B$), Alice needs to be previously authorized by Bob. Thus, Alice follows a three-step protocol to access Bob's content information. For easy of the exposition, we now consider that Alice requests $\mathcal{P}_{Bob}$. However, this is easily extended to a generic content message m. The three-step protocol is illustrated in Figure 3, and described as follows.

1. *Produce Request:* Alice can issue a request in two ways. She either: (i) contacts directly a routing friend of Bob $\mathcal{F}_{Bob_j}$ from $\Gamma^{n}_{(Bob,Alice)}$ (or directly Bob, if he is online); (ii) use a friend $\mathcal{F}_{Alice_i}$ as a trusted entry point of the AN (e.g., Tor). Thus, Alice uses his Λ_{Alice} component to send a request of the form $(\mathcal{P}_{Bob}, r, \psi_{Bob})$, where r is a random value generated for the session, and ψ_{Bob} is a MAC of r using the τ_{Bob}. We observe that this protocol could be extended to use credentials: where Alice request needs to prove she owns some credential issued by Bob.

2. *Authenticate Request:* To access the content, Alice needs to provide a proof that she has τ_{Bob}. Figure 3 sketches a simple method of authentication. $\mathcal{F}_{Bob_j}$ also knows τ_{Bob} and, thus, is able to produce the same MAC output as the one sent by Alice. If the authentication fails, $\Lambda_{\mathcal{F}_{Bob_j}}$ replies $\perp$ to Alice indicating a reject on accessing the content. Otherwise, $\Lambda_{\mathcal{F}_{Bob_j}}$ retrieves and encrypts $\mathcal{P}_{Bob}$ using a hash of the token as the key.

3. *Process Request:* The request is processed by Alice upon received. Note that this step only occurs if the authentication step is valid. If so, Λ_{Alice} decrypts and verifies μ_{Bob} using the hash of τ_{Bob} with r as the key. In this way, Alice retrieves $\mathcal{P}_{Bob}$ in a way that the OSN provider is kept oblivious. Actually, in the prying eyes of the provider $\mathcal{F}_{Bob_j}$ accessed $\mathcal{P}_{Bob}$. Also Bob, as in the normal functionality of the OSN (e.g., Facebook) does not learn who accesses his profile.

Exchanging Messages

We now discuss how to privately send/read a message using our system. For now, we assume that there is only two participants, Alice and Bob, and later we discuss the scenario of multiple recipients. In contrast to the protocol from, this protocol does not make use explicitly of $\mathcal{F}_{Bob_j}$, and connects directly to Λ_{Bob}. Note that for now we assume that both users are online when they exchange a message (in practice, Λ_{Alice} would perform an initial check about peers being online before engaging in a chat). As the goal is to avoid the communication to be observable by the OSN prying eyes, Alice and Bob establish a direct point to point secure channel, such as TLS [18], for the communication. To avoid the fact that NSA is able to decrypt TLS packets as they have access to some end points private keys [34, 35], we encrypt the messages. The protocol is illustrated in Figure 4, for the case where Alice initiates the exchange. Since the cryptographic operations are handled by the components Λ_{Alice} and Λ_{Bob}, this authorization access procedure does not affect the browser.

Let us now address the scenario of multiple recipients, let's say when Alice exchanges messages with Bob, Charlie and Dave. In this case a multiple secure channels. This would create a large overhead of communication. However, privacy often come at a cost, and we argue that this overhead is an unavoidable privacy tradeoff.

Posting Comments

Posting comments involves, usually, multiple readers. As highlighted by our goals, we aim to keep the identity of in-

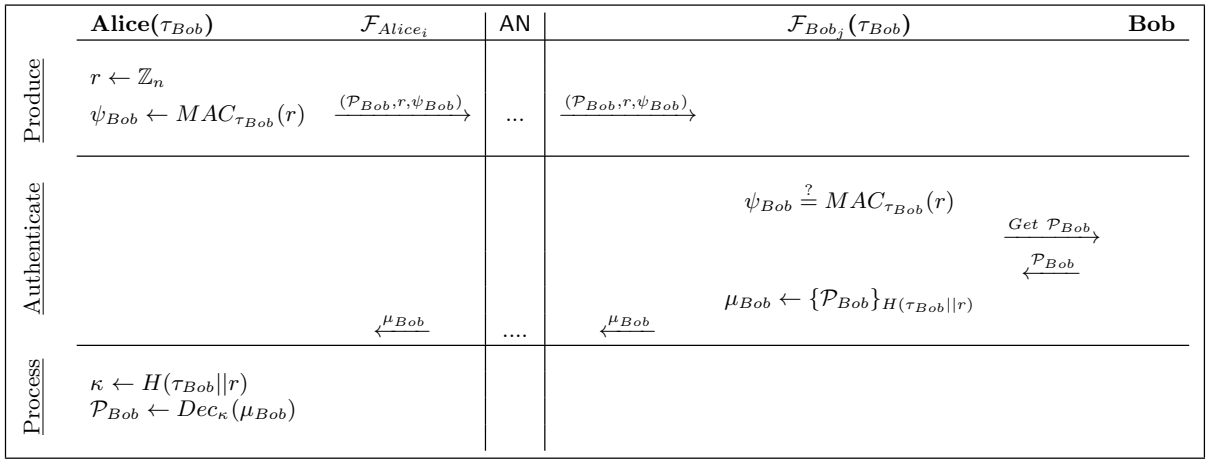

Figure 3: Alice request Bob's profile $\mathcal{P}_{Bob}$ through the routing friends $\mathcal{F}_{Alice_i}$ and $\mathcal{F}_{Bob_j}$. Such that, $i \in \mathcal{R}_{Alice}$ and $j \in \mathcal{R}_{Bob}$.

Alice:	$m_{init} \leftarrow$ ("chat request" $||N$)
Alice $\rightarrow$ Bob:	$\texttt{Enc}_{pk_{Bob}}(m_{init})$, $\texttt{Sign}_{sk_{Alice}}(m_{init})$
Alice $\leftarrow$ Bob:	$\{m_1\}_{\kappa}$, s.t. $\kappa = H(\tau_{Alice}||\tau_{Bob}||N)$
	...
Alice $\leftrightarrow$ Bob:	$\{m_i\}_{\kappa}$

Figure 4: Exchange messages, where κ represent the short lived session key and N a fresh random nonce.

teractions among users anonymous, i.e., keeping the OSN oblivious on who is involved in the interactions. Naturally, Alice cannot post a comment directly to Bob's profile as her identity would be compromised. Also, she cannot use a routing friend $\mathcal{F}$ as it causes an impersonation and thus lead to social issues. With our approach, Alice utilizes Bob to place the comment on her behalf on his profile $\mathcal{P}_{Bob}$. The process can be briefly described as follows. First, Alice sends Bob an encrypted message containing the message and the intended recipient set. Consequently, Bob authenticates Alice and publishes the comment in his wall, to be intended for the limited set of recipients. In this way, Bob can verify the post before being published, edit and publish. For other users comments, our system operates as depicted in Figure 5. Note that, to provide a comment, users must have knowledge of the τ_{Bob}. Each message/comment on the system is composed by the tuple message, user that published the comment and a digital signature for integrity of such user.

Let $\varsigma_u \leftarrow (m, u_{id}, Sign_u(m))$, for $u = \{Alice, Bob, Clark\}$	
Alice $\rightarrow$ Bob:	$c_m \leftarrow \{\varsigma_{Alice}\}_{\kappa}$, s.t. $\kappa = H(\tau_{Bob})$
...	
Bob:	Publishes c_m
...	
Clark $\leftarrow$ Bob:	Request(c_m)
Clark $\rightarrow$ Bob:	$c'_m \leftarrow \{\varsigma_{Alice}||\varsigma_{Clark}\}_{\kappa}$

Figure 5: Post comments process

Currently, the protocol illustrated in Figure 5 does not enforce any kind of access control and its secret κ does not provide forward secrecy. However, later we discuss ways to provide such properties such as the usage of broadcast encryption techniques [4, 30]

5. SECURITY ANALYSIS

We now turn to analyze the security and privacy resilience of our system under the adversary modeled in Section 3. We demonstrate that such an adversary does not learn the interactions occurring, whether working independently or in collaboration with one of the routing friends. Furthermore, we show that a routing friend cannot authenticate himself to access unauthorized content by impersonating other.

User Anonymity: Our system achieves user anonymity when one accesses content of another. This is possible because Alice requests, for example, $\mathcal{P}_{Bob}$ using the protocol in Figure 3. In fact, as aforementioned, tunneling through Tor provides nice security and privacy features, thus, Alice's identity is kept anonymous towards the OSN provider, $\mathcal{F}_{Bob_j}$ and Bob. In fact, on the prying eyes of an adversary such as the OSN provider, $\mathcal{F}_{Bob_j}$ is the one requesting $\mathcal{P}_{Bob}$ or any other content. Note that this process can be done directly through Bob. Besides the fact that Alice reveals τ_{Bob} to $\mathcal{F}_{Bob_j}$ (or Bob) this does not reveals that it is Alice who is requesting the content. This is because τ_{Bob} is also shared with others in $\mathcal{R}_{Bob}$ or $\mathcal{L}_{Bob}$, making the anonymity set large enough for possible match. Also, the same applies on the case that $\mathcal{F}_{Bob_j}$ colludes with the OSN provider.

Communication Anonymity: For an adversary like the OSN provider, it is hard to predict that Alice and Bob are communicating, i.e., exchanging messages or placing comments. In fact, the communication is executed by Λ_{Alice} and Λ_{Bob} using a different channel from the OSN, thus, outside the prying eyes of such adversary. In addition, on the case a stronger adversary (e.g., government) colludes with the Internet Service Providers and listens to the communication, he cannot retrieve the shared secret and thus decrypt the communication. Although, such adversary could infer that Alice and Bob are communicating. This could be solve

by creating a secure channel through a AN, like Tor.

Content Privacy: As all communications are encrypted, only authorized recipients are able to retrieve the content. Further, the authenticity of the message is protected by digital signature. For the authentication, it is hard for an non-authorized user to access any content from Bob. Whereas it is possible for such adversarial user to replay the request message, it is infeasible to decrypt the message as it it encrypted with τ_{Bob}. To avoid the replay attack, a timestamp could be added to the request. Moreover, guessing the τ corresponds to predict the output of the MAC, which can only be done with probability $1/2^n$, where n is the number of bits of the MAC output, which is negligible.

Whereas we tackle several privacy and security issues on the social network, we stress the fact that we do not protect against side-channel attacks, such as timing and correlation attacks.

6. IMPLEMENTATION

To demonstrate the viability of our proposal, we implemented a proof-of-concept prototype of the system proposed in Section 4 as a Firefox plugin, named VF-App.[1] In this section, we describe the architecture (Section 6.1) and the actual steps performed in the communication process (Section 6.2). Later (in Section 6.3), we analyze the performance of our implementation.

6.1 Architecture

The architecture of our VF-App is illustrated in Figure 6. In particular, we can observer two main components: a requester component (VF-Requester) and a routing component (VF-Router). The former executes and processes the requests, whereas, the latter routes, authenticates and retrieves the requested information. Both components are embedded and run as an unique browser extension on the user environment. The working of the components is described in the following.

VF-Requester: this component is used to realize the out-of-band anonymous request information service. VF-Requester handle the request of the user, allowing the choice of which path to take to route the request.

VF-Router: the router component operates as user local server that routes the requested data. Therefore, its role is twofold: (1) from the requester side, e.g., form Λ_{Alice} side, it receives Alice request and forwards it through Tor using Vidalia;[2] (2) when processing a request, e.g., as $\Lambda_{\mathcal{F}_{Bob_j}}$, the authenticator module first verifies that the requester (Alice) is authorized to access the requested data (e.g., Bob profile), then it retrieves the requested information: the retrieve is done using the Facebook authentication token of the user running the extension (i.e., the one of $\Lambda_{\mathcal{F}_{Bob_j}}$ in our example). Each user local server has an associated web address to be contacted, such as IP or domain name base addresses. This allows, for instance, Λ_{Bob} to be reached by Λ_{Alice}.

[1]Source of our implementations is available upon request.
[2]https://www.torproject.org/projects/vidalia.html.en

The current prototype is compatible with Firefox 14+ and could be easily ported to other browsers (e.g., Chrome), as it is written in simple Javascript. Besides its easy installation process, the VirtualFriendShip requires Vidalia for tunneling through Tor. The use of Tor is an important utility as our system uses it to achieve unobservability and anonymity of the communication. Also, our implementation makes use of Polipo[3] to convert the HTTP requests into SOCKS [29], i.e., the protocol operated by Tor. In addition, our communication processes involve simple symmetric-key operations that are executed in Javascript using the Stanford Javascript Crypto Library (SJCL) [38]. Both VF-Client and VF-Server use port 8765 for communication. Finally, we recall from our model that in order to allow Alice to access Bob's profile, at least one $\mathcal{F}_{Bob_j}$ must be online and logged in to Facebook.

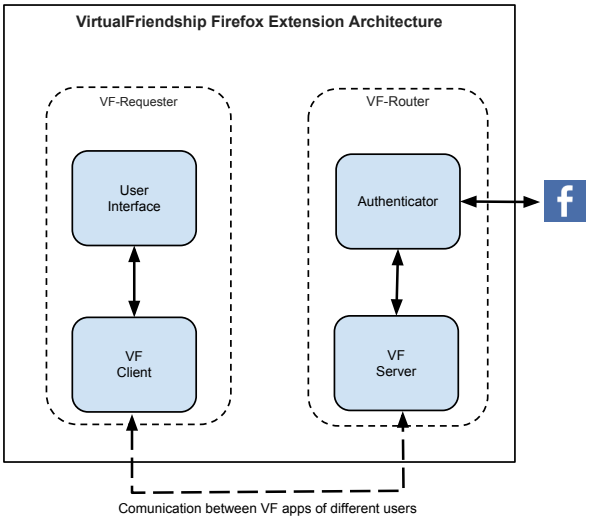

Figure 6: VirtualFriendshp Architecture

To bootstrap the system users establish connections with their friends. Therefore, user u (e.g., Alice) retrieves and uploads a JSON file containing an initial set of information $\mathtt{I}_{u'}$ (i.e., the credentials of user u', e.g., Bob). The set $\mathtt{I}_{u'}$ is composed by: the token $\tau_{u'}$; and a list of the routing friends $\Gamma^n_{(u,u')}$, i.e., friends that u (Alice) can use to reach from u' (Bob). Currently, this exchange is performed automatically using email. However, it could be implemented via other offline channels, such as a USB flash drive. The list of friends is stored locally as a JSON object.

6.2 Communication Process

We have implemented a prototype of VirtualFriendship as Firefox extension (VF-app). In this section, we sketch the steps to access a user profile, assuming that Alice wish to retrieve $\mathcal{P}_{Bob}$. Also, Alice wants to keep this action oblivious towards the OSN provider, as well as to the routing friend of Bob and Bob himself. The VF-app installed in Alice machine will follow our protocol, as depicted in Figure 7.

In particular, VF-app will execute these steps.

1. Select from the list $\Gamma^n_{(Alice,Bob)}$ given by Bob: as an

[3]http://www.pps.univ-paris-diderot.fr/~jch/software/polipo

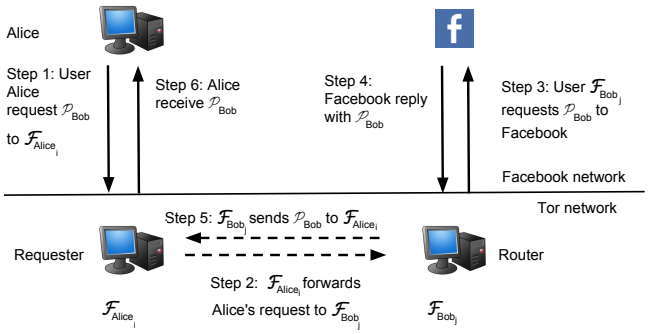

Figure 7: VirtualFriendship Access content Protocol

exit point $\mathcal{F}_{Alice_i}$, and an entry point $\mathcal{F}_{Bob_j}$. The extension automatically encrypts the request using Bob's shared key and sends the the request to $\mathcal{F}_{Alice_i}$, with destination $\mathcal{F}_{Bob_j}$ and requesting $\mathcal{P}_{Bob}$.

2. In order to process the request as soon as it is made, $\mathcal{F}_{Alice_i}$ is required to be online. $\mathcal{F}_{Alice_i}$ receives the request, and forwards it to $\mathcal{F}_{Bob_j}$, using Tor. Please note that this step could be bypassed sending the request directly to $\mathcal{F}_{Bob_j}$ via Tor. However, as prior described, we argue that Alice trusts, socially, $\mathcal{F}_{Alice_i}$ to be her exit point instead of the first node in Tor.

3. The friend $\mathcal{F}_{Bob_j}$ of Bob (which is a entry point to Bob) receives an encrypted request from someone using the shared key of Bob. Hence, $\mathcal{F}_{Bob_j}$ verifies the authenticity of the request. Then, $\mathcal{F}_{Bob_j}$ (which is signed in to Facebook) collects the Facebook token for authentication, and makes a FQL[4] request for Bob's profile to Facebook.

4. Facebook replies to the FQL request from $\mathcal{F}_{Bob_j}$ containing $\mathcal{P}_{Bob}$. We recall that this is possible since B_1 is a Facebook friend of Bob, i.e., $\mathcal{F}_{Bob_j} \in \mathcal{R}_{Bob}$. In fact, on the view of Facebook, it was $\mathcal{F}_{Bob_j}$ who accesses $\mathcal{P}_{Bob}$ (and not Alice).

5. After, $\mathcal{F}_{Bob_j}$ encrypts the Facebook response with Bob shared key, and sends it back to $\mathcal{F}_{Alice_i}$.

6. Finally, $\mathcal{F}_{Alice_i}$ just re-directs the response to Alice. The VF-Requester decrypts the message and displays the $\mathcal{P}_{Bob}$ in the browser window as a regular webpage of the OSN.

We underline that the first and last steps of the protocol are performed by the VF-Requester, where the remainder are executed by the VF-Router component, outside the OSN network. With the above steps, all actions are automated and transparently towards Alice, whereas the OSN provider is kept oblivious toward the actual interaction between Alice and Bob (i.e., that Alice accesses Bob's profile).

6.3 Performance

In order to analyze the practical usability and performance of our system, we focus on measuring two costly factors: the cryptographic overheard added for token protection and authentication of the requested content; and the average communication overheard required for a profile request.

Figure 8 shows the average communication overhead added by the VirtualFriendship when compared to a normal Facebook browsing. We have compared when retrieving just the personal details of the profile (i.e., personal information), such as, name, gender, date of birth and city. With extracting the full profile, i.e., full personal information and recent timeline events. In addition, we analyze the differences when adding Tor. As represented in Figure 8, we can conclude that there is a significant difference on performance. However, we stress that those are milliseconds, and thus, almost negligible on a view of a user.

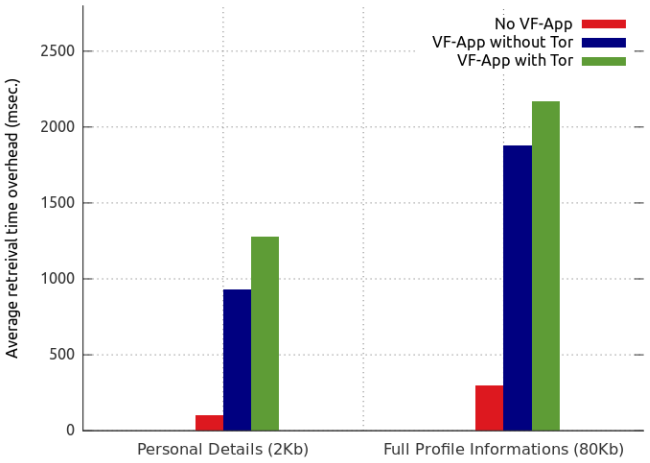

Figure 8: VirtualFriendship time overhead

With respect to the cryptographic overhead, we use AES-CMAC [37] for the MAC implementation and AES-CCM [40] for authenticated symmetric encryption, as they are already available in SJCL [38]. Thus, the authentication process is resumed to MAC implementation which takes about $2msec$. Whereas for the symmetric encryption of the full profile (approximately 80kB), it takes about $10msec$.

7. DISCUSSION AND EXTENSIONS

To the best of our knowledge, VirtualFriendship is the first attempt to construct a hybrid privacy-enhanced extension for current centralized OSNs. However, it is far from tackling all privacy issues in OSNs. Specially, the cases where multiple recipients are involved and access control is required. In this section, we discuss possible extensions to our system regarding access of content management and the use of other privacy-enhanced authentication mechanisms, such as anonymous credentials.

7.1 Access Management

Now we turn to discuss the approach of using different levels of access control. Segregation of information presents an important property for OSN users privacy, as discussed in [6, 39]. Famous OSNs, such as Facebook and Google+, already introduce segregation mechanisms with the use of lists and groups, respectively. Although such mechanisms offer some sort privacy preferences those are coarse and do not protect from the OSN provider.

[4]https://developers.facebook.com/docs/reference/fql/

Currently, our system provides a single token per all connections or per group. For example, that Bob defines the following lists $\mathcal{L}_{Bob}^{Work}$ and $\mathcal{L}_{Bob}^{Family}$, such that $\mathcal{L}_{Bob}^{Work}$ = {Alice, Clark, Dave}, and $\mathcal{L}_{Bob}^{Family}$ = {Mom, Dad, Sister}. This, however, creates a storage overhead and complicates the revocation procedure. With this approach one cannot enforce a more flexible access control per content, e.g., on the case that Alice and Clark are in different groups and the content published should be access by just Alice and Clark. In addition, it does not provide transparency, as Alice is not aware of who else is in $\mathcal{L}_{Bob}^{Work}$ besides $\Gamma_{(Alice,Bob)}^{n}$.

From existing solutions tackling content privacy, Scramble! [7] is the only that addresses such issue, employing access control per content with anonymous broadcast encryption (ABE) [4, 30]. Whereas the recipient set is kept anonymous to outsiders, it is also anonymous to the insiders (i.e., towards the recipients of the content). Thus, the access control list employed is not transparent. For instance, Alice is in the set but she does not know who else is in the set. Furthermore, such approach although useful to enforce flexible access control per content.

7.2 Anonymous Credentials

Authentication represents an important functionality of our system. As described in the security analysis, our protocol does not provide forward secrecy. In fact, an attacker can replay the request although he cannot decrypt it. The authentication tokens provided (e.g., Alice provides τ_{Alice} to Bob) could be generated in collaboration, using, for example, an authenticated Diffie-Helman key agreement protocol. However, such solution would not provide anonymity as each user would have an unique token and thus an unique identifier.

Another possibility to perform authentication to a certain content is to use anonymous credentials [10, 12]. Such solution, yet more costly efficient, allows users to prove knowledge of some property in an anonymous way. For instance, Alice can prove to $\mathcal{F}_{Bob_j}$ or Bob himself that she is eligible to access the requested content without disclosing her identity. To setup such system, however, the initialization protocol would become more complicated. Actually, by following the same use case, Bob should issue a new credential to Alice and indicate which $\mathcal{F}_{Bob_j}$ can act as verifiers. Consequently, the τ_{Bob} would become a one-time token generated as a proof of knowledge of the credential. In this way, Alice benefits the nice privacy and security properties, such as anonymity and forward secrecy, during the system operation with an efficiency tradeoff. In addition, Bob can perform credential revocation in a more efficient way.

7.3 Voting actions

Voting actions, such as "like" in Facebook, are simple to perform and extensive used actions in OSNs. However, such actions allow an adversary to compromise the privacy of users. In particular, such adversary can compute the weights and directly determine the strength of friendships and associate common interests. Thus, it is hard to protect users identity when such voting actions are performed. Whereas addressing this question is beyond the scope of this paper, an interesting solution could be the use of e-voting techniques with double spending protection, e.g., [11]. However, such solution requires an extra entity to act as a bank and issue and manage coins. Consequently, as aforementioned, effective means of addressing this issues are beyond this paper's scope.

7.4 Mobile Extension

With the enormous growth on usage of mobile devices, such as tablets and smartphones, mobile users represent the majority of traffic in OSNs [13]. Currently, our implementation is compatible with Firefox 14.x, however, the low overhead of VirtualFriendship makes it appropriate to such constrained devices (e.g., AES-CCM encryption takes about $50msec$ on constrained laptop). In fact, the mobile setting can represent a asset to the availability of routing friends. Nowadays such devices allow users to be constantly online, thus, as each user in the VirtualFriendship system runs a local server this allows such server to be also continuously online.

8. CONCLUSION

This paper presents a solution to mitigate the problem of privacy as hiding user's interactions when using centralized OSNs. We adhere to an adversary that is able to link users interactions and further associate the strength of such connections, such as the OSN provider. We analyzed the privacy risks on current OSNs and proposed a system that allows end-peers to interact anonymously within the OSNs. We present, to the best of our knowledge, the novel approach of *routing friends*, where we abuse the definition of trust in social interactions from [17]. Furthermore, we have implemented our system as a proof of knowledge prototype to demonstrate the feasibility of our approach. Later, we evaluated and show that the overhead introduced by the solution is limited and, thus, we believe to be negligible to users.

Our current solution, as discussed throughout this paper, did not plan or solve all the privacy issues on OSNs. Thus, we foresee potential future work and open questions. In particular, extensions with the implementation of comments, revocation of users without re-distribution of access tokens, and address anonymity of actions such as "like" in Facebook. In addition, we aim to extend the proof of knowledge implementation with all available operations into a real world open source application. Also, as a large number of users use their mobile devices to connect to OSNs, a mobile version of the system becomes attractive.

9. REFERENCES

[1] A. Acquisti, B. Van Alesenoy, E. Balsa, B. Berendt, D. Clarke, C. Diaz, B. Gao, S. Gurses, A. Kuczerawy, J. Pierson, F. Piessens, R. Sayaf, T. Schellens, F. Stutzman, E. Vanderhoven, and R. De Wolf. The SPION Project. `https://www.cosic.esat.kuleuven.be/publications/article-2077.pdf`.

[2] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin. Persona: an online social network with user-defined privacy. *SIGCOMM Computing Communication Review*, 39(4):135–146, 2009.

[3] E. Balsa, C. Troncoso, and C. Díaz. A metric to evaluate interaction obfuscation in online social networks. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 20(6):877–892, 2012.

[4] A. Barth, D. Boneh, and B. Waters. Privacy in encrypted content distribution using private broadcast encryption. In *In Financial Cryptography âĂŹ06*, page 2006. Springer. LNCS, 2006.

[5] F. Beato, I. Ion, S. Capkun, B. Preneel, and M. Langheinrich. For some eyes only: protecting online information sharing. In E. Bertino, R. S. Sandhu, L. Bauer, and J. Park, editors, *CODASPY*, pages 1–12. ACM, 2013.

[6] F. Beato, M. Kohlweiss, and K. Wouters. Enforcing access control in social network sites. *Hot Topics in Privacy Enhancing Technologies (HotPETS)*, 2009.

[7] F. Beato, M. Kohlweiss, and K. Wouters. Scramble! your social network data. In S. Fischer-Hübner and N. Hopper, editors, *PETS*, volume 6794 of *Lecture Notes in Computer Science*, pages 211–225. Springer, 2011.

[8] J. Bonneau, J. Anderson, R. Anderson, and F. Stajano. Eight friends are enough: social graph approximation via public listings. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, SNS '09, pages 13–18, New York, NY, USA, 2009. ACM.

[9] D. Boyd and N. Ellison. Social Network Sites: Definition, History, and Scholarship. *Journal of Computer-Mediated Communication*, 13(1), 2008.

[10] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich. How to win the clone wars: efficient periodic n-times anonymous authentication. Cryptology ePrint Archive, Report 2006/454, 2006. `http://eprint.iacr.org/`.

[11] J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact e-cash. In *In EUROCRYPT, volume 3494 of LNCS*, pages 302–321. Springer-Verlag, 2005.

[12] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[13] J. Constine. Mobile as it starts sharing user counts by country. Aug 13, 2013. `http://techcrunch.com/2013/08/13/facebook-mobile-user-count/`. Accessed on Sept. 21, 2013.

[14] M. Conti, A. Hasani, and B. Crispo. Virtual private social networks. In R. S. Sandhu and E. Bertino, editors, *CODASPY*, pages 39–50. ACM, 2011.

[15] L. A. Cutillo, R. Molva, and M. Önen. Safebook: A distributed privacy preserving online social network. In *WOWMOM*, pages 1–3. IEEE, 2011.

[16] L. A. Cutillo, R. Molva, and T. Strufe. Safebook: Feasibility of transitive cooperation for privacy on a decentralized social network. In *WOWMOM*, pages 1–6. IEEE, 2009.

[17] G. Danezis, C. Diaz, C. Troncoso, and B. Laurie. Drac: an architecture for anonymous low-volume communications. In *Proceedings of the 10th international conference on Privacy enhancing technologies*, PETS'10, pages 202–219, Berlin, Heidelberg, 2010. Springer-Verlag.

[18] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), 2008.

[19] R. Dingledine, N. Mathewson, and P. Syverson. Tor: the second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[20] C. Dwyer, S. R. Hiltz, and K. Passerini. Trust and privacy concern within social networking sites: A comparison of facebook and myspace. In J. A. Hoxmeier and S. Hayne, editors, *AMCIS*, page 339. Association for Information Systems, 2007.

[21] J. Dwyer. Four nerds and a cry to arms against Facebook. May 11, 2010. `http://www.nytimes.com/2010/05/12/nyregion/12about.html`. Accessed on Sept. 3, 2012.

[22] Facebook and your privacy: Who sees the data you share on the biggest social network? Consumer Reports magazine, June 2012. `http://www.consumerreports.org/cro/magazine/2012/06/facebook-your-privacy/index.htm`. Accessed on Sept. 6, 2012.

[23] L. Fang and K. LeFevre. Privacy wizards for social networking sites. In *WWW*, 2010.

[24] M. J. Freedman and R. Morris. Tarzan: a peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 193–206, New York, NY, USA, 2002. ACM.

[25] S. Guha, K. Tang, and P. Francis. Noyb: privacy in online social networks. In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 49–54, New York, NY, USA, 2008. ACM.

[26] S. F. Gurses. *Multilateral Privacy Requirements Analysis in Online Social Networks*. PhD thesis, Katholieke Universiteit Leuven, 2010. Bettina Berendt and Bart Preneel (promotors).

[27] S. F. Gurses and C. Diaz. Two tales of privacy in online social networks. *IEEE Security & Privacy*, x(x):8, 2013.

[28] C. Jernigan and B. F. T. Mistree. Gaydar: Facebook friendships expose sexual orientation. *First Monday*, 14(10), 2009.

[29] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. SOCKS Protocol Version 5. RFC 1928 (Proposed Standard), March 1996.

[30] B. Libert, K. G. Paterson, and E. A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In M. Fischlin, J. Buchmann, and M. Manulis, editors, *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 206–224. Springer, 2012.

[31] W. Luo, Q. Xie, and U. Hengartner. Facecloak: An architecture for user privacy on social networking sites. In *CSE'09*, pages 26–33, Washington, DC, USA, 2009. IEEE Computer Society.

[32] P. Mittal, M. Wright, and N. Borisov. Pisces: Anonymous communication using social networks. In *NDSS*, 2013.

[33] A. Mohaisen and Y. Kim. Dynamix: anonymity on dynamic social structures. In *Proceedings of the 8th ACM SIGSAC symposium on Information, computer*

and communications security, ASIA CCS '13, pages 167–172, New York, NY, USA, 2013. ACM.

[34] J. L. Nicole Perlroth and S. Shane. NSA able to foil basic safeguards of privacy on web. Sept. 5, 2013. `http://www.nytimes.com/2013/09/06/us/nsa-foils-much-internet-encryption.html`. Accessed on Sept. 6, 2013.

[35] W. Post. NSA slides explain the PRISM data-collection program. June 6, 2013. `http://www.washingtonpost.com/wp-srv/special/politics/prism-collection-documents/`. Accessed on Sept. 6, 2013.

[36] C. Riederer, V. Erramilli, A. Chaintreau, B. Krishnamurthy, and P. Rodriguez. For sale : your data: by : you. In *Proc. of ACM HotNets-X 2011*, pages 13:1–13:6.

[37] J. Song, R. Poovendran, and J. Lee. The AES-CMAC-96 Algorithm and Its Use with IPsec. RFC 4494 (Proposed Standard), 2006.

[38] E. Stark, M. Hamburg, and D. Boneh. Symmetric Cryptography in Javascript. In *ACSAC*, 2009.

[39] B. van den Berg and R. Leenes. Keeping up appearances: Audience segregation in social network sites. In S. Gutwirth, Y. Poullet, P. D. Hert, and R. Leenes, editors, *Computers, Privacy and Data Protection*, pages 211–231. Springer, 2011.

[40] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). RFC 3610, Sept. 2003.

[41] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao. User interactions in social networks and their implications. In W. Schröder-Preikschat, J. Wilkes, and R. Isaacs, editors, *EuroSys*, pages 205–218. ACM, 2009.

[42] H. Wu and B. Preneel. AEGIS: A Fast Authenticated Encryption Algorithm. In *Selected Areas in Cryptography*, 2013. To appear.